HTML5 ウェブストレージ記憶のメインメモリからの獲得

松本晋一[†1†2]　　櫻井幸一[†1†2]

近年 Web ブラウザは，HTML 5 標準化の進展を背景として，種々のアプリケーションが動作するプラットフォームとして成長しており，これにより，ブラウザ上にネットワーク上でのユーザの活動の履歴を反映した情報が集約されることになり，これらの情報は重要なエビデンスとなるものと考えられる．HTML5 はスマートフォンのプラットフォームとしても有望視されており，HTML5 の固有属性を対象としたフォレンジクスはモバイルフォレンジクスにおいて今後重要となるものと考えられる．本論文では，Web ブラウザ動作後の端末内のメモリイメージから， HTML5 ウェブストレージ内に記憶された情報を獲得する実験を行った結果について報告する．

Acquisition of HTML5 Web Storage from Main Memory

Shinichi Matsumoto[†1†2]　Kouichi Sakurai[†1†2]

Recent years, web browser is growing as a platform to execute various applications. This movement is motivated by the standardization of HTML 5. In this situation, much more activity records of user are aggregated on the web browser and these records may contain significant evidence for forensics. HTML5 is promised as a platform for smartphones. Therefore HTML specific attribute will be useful as evidence for forensics. In this paper, we report the result of experiment that acquire the main memory image within terminal and extract the web storage data as an evidence of browsing activity.

1. はじめに

1.1 デジタル・フォレンジック

デジタル・フォレンジックは，近年特にその重要性が高まりつつあり，社会的にもその価値，重要性に関する認識が広まりつつある．デジタル・フォレンジックにおいては，企業，個人等の活動に関する情報を保全し，正確かつ分かりやすく開示することが重要となる．近年デジタル・フォレンジックに対する要請は，法制度からだけではなく経営管理などからも高まっている．これは，コンピュータやインターネットの普及に伴い，個人の活動や企業活動が様々な形で電子的に行われる状況に対し，刑事的/民事的な訴訟の証拠を，電子的な活動の記録に依拠せざるを得ないためである．

[1]では，デジタル・フォレンジックの動向として，以下の分野での重要性が今後増すものと予想している．

(1) 不正侵入などの証拠性だけでなく，財務会計情報などの情報改ざんや，個人情報/機密情報の漏洩，詐欺などの不正行為が無かったかの証拠性．今後グローバル化の進展に伴い，日本企業においても透明性の確保，アカウンタビリティが求められるケースが増えることが予想され，経営管理の観点からデジタル・フォレンジックを捉える必要性が増すものと考えられる．

(2) 訴訟を起こす側だけでなく，訴訟を起こされる側となる場合に備え，証拠を確保する必要性．日本企業においてもこのような状況での証拠提出が求められる法律の整備が進んでおり，今後重要な課題となるものと考えられる．

(3) 従来，デジタルデータを扱うものとして，サーバやパーソナルコンピュータを考えていたが，今後は重要な情報が携帯端末，特に現在市場を拡大し続けているスマートフォンでも扱われており，これらを対象とする必要性は，今後ますます高まるものと考えられる．

本研究では，このような予想をもとに，モバイルフォレンジクス，特に今後スマートフォンに代表される携帯端末を対象としたモバイルフォレンジック技術に応用可能な実験を行う．

1.2 モバイルデバイスのフォレンジック

モバイルデバイスを対象としたフォレンジックについては，すでに活用事例があるが，今後その必要性はますます高まることが予想される．

前節に述べたように，モバイルデバイスの主流は，従来の携帯電話からスマートフォンやタブレットに移りつつあり，スマートフォン/タブレットを特徴付ける，インストール可能なアプリケーションの実行環境は，プラットフォームが備えるバイナリ実行環境，あるいは HTML/JavaScript 実行環境に依存している．バイナリ実行環境を持つスマートフォン/タブレットプラットフォームとしては Android, iOS や Tizen などがあり，HTML/JavaScript 実行環境のみを備えるプラットフォームは Firefox OS, Ubuntu Touch などがある．

バイナリ実行環境を備えるスマートフォン/タブレットプラ

†1 公益財団法人 九州先端科学技術研究所
Institute of Systems, Information Technologies and Nanotechnologies (ISIT)
†2 九州大学 システム情報科学府 情報学専攻
Department of Informatics, Faculty of Information Science and Electrical Engineering, Kyushu University

ⓒ2012 Information Processing Society of Japan

ットフォームについても，プラットフォーム間でのアプリケーションの移植性を考慮し，HTML/ JavaScript に基づくアプリケーション実行環境の重要性は今後増していくことが予想される．また，ナビゲーション端末や情報家電のソフトウェアも，同様に HTML/ JavaScript ベースとなると予想され, 今後は Web ブラウザ/ツールキット内の HTML/JavaScript ランタイムが処理する情報に対するフォレンジック技術が必要とされると考えられる．

1.3 HTML5 生成物のフォレンジック

HTML 5 [2]においては, Web アプリケーション開発を助けるために様々な機能が追加されており，3 章で述べる種々の拡張により，Web ブラウザ側で，音声や動画などのマルチメディア情報を直接扱える一方, 従来の cookie[3]よりも Web ブラウザ側で大幅に制約の少ない利用法が可能な Web storage[4]により, アプリケーション次第で Web ブラウザ側に種々の情報を記録することが可能となる．考えられる用途としては,

- メールアプリケーション内での，メール本文および送信元，宛先などのヘッダ情報.
- フォトアルバムアプリケーション内での，写真/画像データや撮影時刻，撮影場所などのメタデータ情報.
- オフィスアプリケーション内での，文書データや編集履歴に関する情報.
- ゲームアプリケーション内でのアカウント情報やプレイ履歴情報.

モバイルデバイスのフォレンジックにおいては，これらのデータを取り扱い可能とすることで，適用可能な領域が大幅に拡張されることが想定される．

2. 先行研究

Web ブラウザを対象としたデジタル・フォレンジックに関する研究[5][6]においては，Web ブラウザの残したログファイル，また一部の Web ブラウザにおいて, ブラウザ内情報専用のストレージとして用いられている SQLite を直接の調査対象とするものとしている．

一方，[7]においては，各種 Web ブラウザ(Mozilla Firefox, Google Chrome, Internet Explorer, Apple Safari)のプライベートブラウジングモード, およびポータブル Web ブラウザを用い各種エビデンスの取得を試みている．プライベートブラウジングモードについては Web ブラウザによって以下のように異なる名称を持つ.

- MS Internet Explorer　InPrivate Browsing
- Google Chrome　Incognito mode
- Mozilla Firefox　Private Browsing
- Apple Safari　Private Browsing

またポータブル Web ブラウザとしては, 以下を調査対象としている.

- Mozilla Firefox Portable
- Google Chrome Portable
- Opera Portable

調査対象として，ファイルシステム上のエビデンスのみならず，メモリイメージを取得し，解析対象としている．[7]において取得に成功したエビデンスは，ブラウザにより異なるが，以下のようなものがある.

- プライベートブラウジングモードによる閲覧を示すインディケータ
- 閲覧履歴
- 画像イメージデータ
- ユーザーネーム/e-mail アカウント

3. HTML 5 概要

HTML 5 は現在 W3C および WHATWG[8]にて標準策定中であり，2014 年の制定が予定されている.

3.1 HTML 5 の特徴的要素

HTML5 においては, 旧版に対して様々なエレメント, 属性, API が追加されており[9]，すでにいくつかのセキュリティ上の懸念が提出されている[10][11]. 新規エレメントの代表としては, 以下がある.

- 文書構造を定義するためのエレメント(section など).
- コンテンツ構造を定義するためのエレメント(article, main, aside など)
- マルチメディアコンテンツのためのエレメント(audio, video など).
- 鍵ペア生成のためのエレメント(keygen)

その他，旧版で定義されたエレメントや属性の変更，また非推奨指定も行われている.

また，これらの変更に関連し，以下に示す新規 API が定義されている．これらについては W3C において独立した文書で定義されている API を含む.

- マルチメディアコンテンツのためのエレメント(video, audio エレメント)が持つ API
- フォーム制約検証 API (setCustomValidity()メソッド)
- ユーザが起動できるコマンド(command エレメント)に関する API
- オフライン Web アプリケーションのためのアプリケーションキャッシュに関する(manifest 属性など)API.
- Web アプリケーションが自身を特定のプロトコル

ⓒ2012 Information Processing Society of Japan

(registerProtocolHandler()メソッド), 特定のメディアタイプに対し登録する(registerContent Handler()メソッド)API.

- Contenteditable 属性と共に用いるコンテンツ編集 API
- draggable 属性と共に用いる Drag & Drop API.
- スクリプトがナビゲーション等を行うための文書の URL コンポネントを提供する(Location インタフェース)API.
- スクリプトがセッション履歴内の文書の URL を変更可能とする(History インタフェース)API.
- base64 エンコード/デコードのための API (atob())/(btoa()).
- タイマベースのコールバックで用いる API(setTimeout(), setInterval()).
- ユーザへのプロンプトを出すための API(alert(), confirm(), prompt(), showModalDialog()).
- 印刷に関する API(print()).
- 検索プロバイダに関する API (AddSearch Provider(), IsSearchProviderInstalled())
- マイクロデータに関する API(itemscope, item type 属性など)[12].
- イミディエイトモードのビットマップグラフィクスのための API(canvas エレメントの 2d コンテクスト)[13]
- Web メッセージ処理のための API (postMessage(), MessageChannel オブジェクト，MessagePort オブジェクト)[14].
- スクリプトのバックグラウンドで実行のための API(Worker オブジェクト，SharedWorker オブジェクト)[15].
- クライアント側のストレージのための Web storage API (local storage，session storage).
- クライアント/サーバの双方向通信のための API (WebSocket オブジェクト)[16].
- サーバ側からクライアント側へ，データのプッシュを行うための, サーバ送信イベント API (EventSource)[17].

3.2 Web storage

HTML 5 において追加された機能の一つである Web storage は，クライアント側にデータを記録する機構であり，これまで広く用いられてきた cookie に類似しているが，記憶容量の制限などが大幅に緩和され, これまでとは異なる応用が期待される.

記憶容量としては，オリジン(3.2.4 節参照)単位で管理が行われ，5MB 以上の割り当てが推奨されている．ただし Web ブラウザ全体での容量制限などが課される場合もあり得る.

3.2.1 Web storage の API

Web storage はキー/バリュー型の API であり，キーとバリューの対からなるエントリのリストからなる記憶構造を持つ. Web storage にアクセスするためのメソッドとして，以下が定義されている.

- setItem(key, value)
  リスト内から引数の key をもつエントリの存在を調べ, もし存在しなければ，key に対応する値を value としてリストに追加する.
  もし key を持つエントリが存在すれば，当該 key に対応する値を value に更新する.
  value を追加できない場合は例外を発生させる.
- getItem(key)
  リスト内から引数の key に対応する value を返す. リスト内に key に対応するエントリが存在しない場合は，null を返す.
- removeItem(key)
  リスト内に引数の key に対応するエントリが存在すれば削除する．対応するエントリが存在しない場合は何もしない.
- clear()
  リストを空にし，全ての key/value の対を削除する．もしリストが空であった場合は，何もしない.
- key(n)メソッド
  リスト内の n 番目の key を返す．リスト内のエントリの順序は，エントリの追加/削除がない限り保存される．n がリスト長以上の場合，null を返す.
- length 属性
  key/value 対のエントリ数を返す.

Web storage には session storage と local storage の二種類がある. 両者の違いは，領域内のデータの保存期間である．保存期間の違いにより，両者はブラウザ内で異なる実装により管理されているものと考えられる.

3.2.2 Session storage

Session storage 領域のデータの保存期間については，明示的な削除要求を行う場合や, ブラウザのストレージ空間の制限や, セキュリティ上の理由に抵触したりしない限り．データにアクセス可能なスクリプトの実行中はデータが存在する．トップレベル閲覧コンテクストが破壊された場合(即ち, ユーザが永遠にアクセス不可能となった場合)には，Session storage 領域内に記憶されたデータは破棄される.

トップレベル閲覧コンテクストを持つ閲覧コンテクストにおいて Document オブジェクトが生成された時，ブラウザは当該トップレベル閲覧コンテクストが当該 Document のオリジン

ⓒ2012 Information Processing Society of Japan

に対応する Session storage 領域の存在を確認する．存在しなかった場合は，当該文書のオリジンに対応する新規ストレージ領域が生成され，これが Document に対応する Session storage 領域となる．

3.2.3 Local storage

Local storage 属性がアクセスされた場合，ブラウザは以下の処理を行う(Storage オブジェクトの初期化).

- 要求がポリシイ (例えばブラウザが永続的記録を許可しない設定した場合)の違反を引き起こす場合は，Storage オブジェクトを返すのではなく SecurityError 例外を投げる．
- Document のオリジンが scheme/host/port のタプル形式でない場合，SecurityError 例外を投げ，このステップを中断する．
- ブラウザが，Window オブジェクトの Document のオリジンに対応する local storage 領域の割り当てを確認する．割り当てられていない場合には，オリジンに対応する新規の記憶領域を生成する．

オリジンの local storage 領域に対応する Storage オブジェクトを返す．

3.2.4 オリジン

オリジン[18]は，Web ブラウザがコンテンツを隔離する単位である．当該概念はコンテンツの守秘性と完全性を確保するために導入されており，Web storage においては，ストレージ内容の隔離と，容量の管理に用いられる．

オリジンは

- プロトコルスキーム
- 完全に修飾されたホスト名
- ポート番号

の組で構成され，この組が合致する場合同一オリジンと見なされる．

オリジン毎に Web storage 領域は隔離され，任意のオリジンから書き込まれた Web storage に対し，異なる任意のオリジンから読み込むことはできない．またあるオリジンで使用した Web storage 領域のサイズが，Web ブラウザの定める最大容量を超えた場合は，追加の書き込みに制限を受ける．

4. 実験概要

4.1 実験目的

本実験では，HTML 5 において，デジタル・フォレンジックの対象として特に有効と考えられる Web storage に記録される情報を，ファイルシステム以外から獲得する方法について検証する．

Web ブラウザから Web storage へ書き込まれた内容を，Web ブラウザ終了後，メインメモリのメモリイメージから獲得可能か否かを検証し，メモリイメージを対象としたフォレンジクスの可能性を探る．

Web ブラウザとしては，デスクトップ，タブレット及びコンソール用として市場でのシェア上位から，Google Chrome, Microsoft Internet Explorer, Mozilla Firefox を対象とする[19].

4.2 実験環境

本稿における実験に用いた環境を以下に示す．また実験に用いた HTML/JavaScript コード抜粋については本稿末尾の付録に示す．

- ホスト環境
  Microsoft Windows 7 Home Premium (64bit)
  主記憶 8GB
- 仮想環境
  Oracle VirtualBox 4.3.2
- ゲスト OS
  Microsoft Windows 7 Ultimate (64bit)
  CPU シングルコア割り当て
  主記憶 4GB 割り当て
- Web ブラウザ
  Mozilla Firefox 25.0.1
  Google Chrome 31.0.1650.63m
  Microsoft Internet Explorer 10.9.9200.16750
- メモリダンプツール
  AccessData FTKImager 3.1.4.6(図 1)[20]

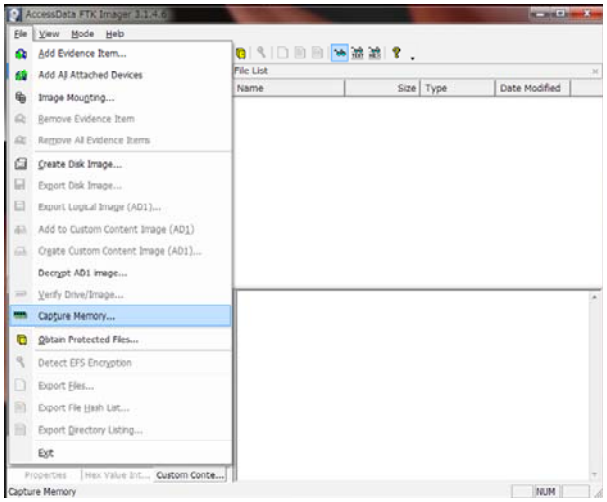

図 1 FTKImager の画面イメージ

ⓒ2012 Information Processing Society of Japan

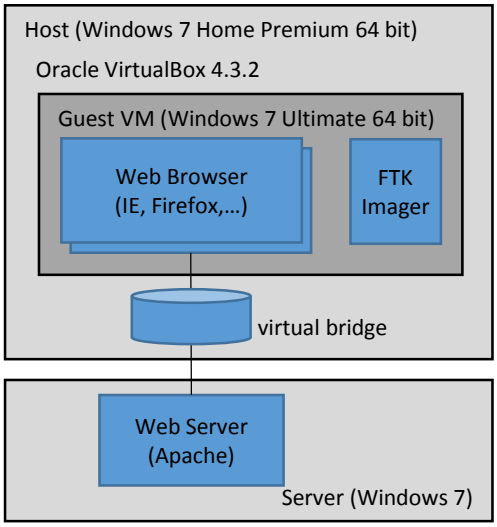

図 2 実験システムの構成

4.3 実験システム構成

実験システムの構成を図 2 に示す．フォレンジック対象となる Web ブラウザは，ハイパーバイザ上のゲスト OS 上で動作させている．

またメモリイメージの取得を行う FTK Imager も，ゲスト OS 上で動作する．

Web サーバはネットワークを介した端末内で動作する．ブラウザが動作するゲスト OS とは，ハイパーバイザの仮想ブリッジモードで接続する．

4.4 実験手順

Web サーバからロードされた Web ページに包含されるメッセージを，Web ページ内 JavaScript コードによりクライアント内 Web storage に書き込む．

この時，Web storage に記録する内容がネットワーク上，また Web ページ上に陽に出現するのを防ぐため，Web ページ内のメッセージは ROT13 にて簡単な変換を施したものを収容する．クライアントでは当該メッセージを再度 ROT13 変換(デコード)し，これを Web storage に変換して記録する(図 3).

各実験においては，クライアント端末においてゲスト VM を起動した後，Web ブラウザを起動し，サーバ上の Web ページをロードする．Web ページのロード後，Web storage へのメッセージの書き込みが行われる．

Web ページ表示を確認した後，ブラウザを終了させ，FTK Imager にてゲスト VM 全体のメモリイメージを取得し，ゲスト VM を終了する．各ブラウザについて，この手順を適用する．

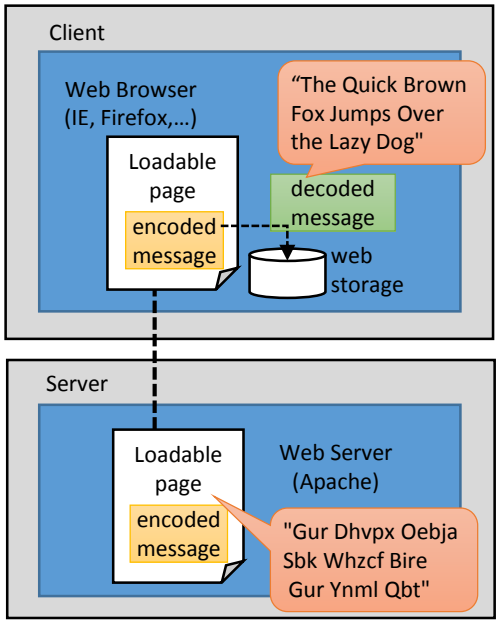

図 3 メッセージの変換と記録

5. 実験結果

本章では，メモリイメージ取得実験の結果について，Web ブラウザの種別，またストレージの Local storage, Session storage の別毎に示す．

5.1 Google Chrome

Google Chrome 実行後のメモリイメージからは, Local storage, Session storage 両方について，オリジン，キー，バリューの各値をメモリイメージから確認できず, エビデンス取得に失敗した．

5.2 Mozilla Firefox

5.2.1 Local storage

Mozilla Firefox 実行後のメモリイメージ内から，Local storage 記憶のオリジン，キー，バリューの各値の取得に成功した．メモリイメージ上の 2 箇所に出現しており，記録エビデンスの取得例を図 4 に示す．

図の上部はバイナリダンプ値であり，文字列として抽出，整形したものを図下部に示す．

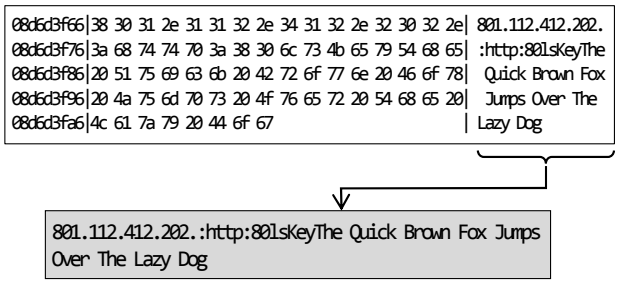

図 4 Firefox の Local storage エビデンス取得例

ⓒ2012 Information Processing Society of Japan

メモリイメージ上，オリジン情報のうち，アドレス(ホスト名)は逆順に収容されていることがわかる．このため，オリジン値が既知であれば，アドレス部を変形し検索することでメモリイメージ上，機械的に検索可能である．またキー値が既知であれば，キー値を検索することで検索可能である．しかしキー値及びバリュー値はメモリイメージ上でデリミタ無しで連結して収容されており，キーとバリューの少なくともどちらか既知でなければ，値を分離できない．

5.2.2 Session storage

Mozilla Firefox 実行後のメモリイメージ内から，Session storage 記憶のオリジン，キー，バリューの各値の取得に成功した．メモリイメージ上の 5 箇所に出現しており，記録エビデンスの取得例を図 5 に示す．

図上部はダンプ値であり，文字列として抽出，整形したものを図下部に示す．

メモリイメージ上，オリジン，キー，バリューの各値が JSON 形式で収容されていることがわかる．当該形式でエンコードされていることから，オリジン値もしくはキー値が既知であれば，機械的に検索し，バリュー値を獲得可能であることが判る．

5.3 Microsoft Internet Explorer

5.3.1 Local storage

Microsoft Internet Explorer 実行後のメモリイメージ内から，localStorage 記憶のキー，バリューの各値の取得に成功した．メモリイメージ上の 2 箇所に出現しており，記録エビデンスの取得例を図 6 に示す．

図上部はダンプ値であり，文字列として一部を抽出，整形したものを下部に示す．

メモリイメージ上でキー，バリューの各値は，その他の情報と共に DOM 形式で収容されていることが確認できる．このことから，キー値が既知であれば，機械的な検索によりバリュー値も獲得可能であることが判る．

```
06e6541e6|7b 22 68 74 74 70 3a 2f 2f 32 30 32 2e 32 31 34| {"http://202.214
06e6541f6|2e 32 31 31 2e 31 30 38 22 3a 7b 22 73 73 4b 65| .211.108":{"ssKe
06e654206|79 22 3a 22 54 68 65 20 51 75 69 63 6b 20 42 72| y":"The Quick Br
06e654216|6f 77 6e 20 46 6f 78 20 4a 75 6d 70 73 20 4f 76| own Fox Jumps Ov
06e654226|65 72 20 54 68 65 20 4c 61 7a 79 20 44 6f 67 22| er The Lazy Dog"
06e654236|7d 7d                                           | }}
```

```
{"http://202.214.211.108":
    {"ssKey":"The Quick Brown Fox Jumps Over The Lazy Dog"}
}
```

図 5 Firefox の Session storage エビデンス取得例

```
083a895ae|3c 69 74 65 6d 20 6e 61 6d 65 3d 22 6d 79 2d 6e| <item name="my-n
083a895be|6f 74 65 22 20 76 61 6c 75 65 3d 22 47 75 72 20| ote" value="Gur
083a895ce|44 68 76 70 78 20 4f 65 62 6a 61 20 53 62 6b 20| Dhvpx Oebja Sbk
083a895de|57 68 7a 63 66 20 42 69 72 65 20 47 75 72 20 59| Whzcf Bire Gur Y
083a895ee|6e 6d 6c 20 51 62 74 22 20 6c 74 69 6d 65 3d 22| nml Qbt" ltime="
083a895fe|38 39 31 30 33 34 34 30 30 22 20 68 74 69 6d 65| 891034400" htime
083a8960e|3d 22 33 30 33 33 39 32 32 35 22 20 2f 3e 3c 69| ="30339225" /><i
083a8961e|74 65 6d 20 6e 61 6d 65 3d 22 6c 73 4b 65 79 22| tem name="lsKey"
083a8962e|20 76 61 6c 75 65 3d 22 54 68 65 20 51 75 69 63|  value="The Quic
083a8963e|6b 20 42 72 6f 77 6e 20 46 6f 78 20 4a 75 6d 70| k Brown Fox Jump
083a8964e|73 20 4f 76 65 72 20 54 68 65 20 4c 61 7a 79 20| s Over The Lazy
083a8965e|44 6f 67 22 20 6c 74 69 6d 65 3d 22 35 30 33 31| Dog" ltime="5031
083a8966e|37 35 37 32 38 22 20 68 74 69 6d 65 3d 22 33 30| 75728" htime="30
083a8967e|33 34 30 34 35 37 22 20 2f 3e                   | 340457" />
```

```
<item name="lsKey" value="The Quick Brown Fox Jumps Over The Lazy
 Dog" ltime="50315728" htime="30340457" />
```

図 6 Internet Explorer の Local storage エビデンス取得例

更にこの際，変換前のメッセージも隣接して収容されており，ブラウザ内では同一 DOM ツリー内で管理されているものと推測される．

5.3.2 Session storage

Microsoft Internet Explorer 実行後のメモリイメージから，Session storage 記憶について，オリジン，キー，バリューの各値のエビデンス取得には失敗した．

6. 考察

6.1 エビデンスの有効性

メモリイメージを対象としたフォレンジクスを行う場合，端末全体のメモリイメージサイズは数 GB のオーダとなり，イメージ中からエビデンスとなる情報を機械的に検索可能であるか否かは問題であり，またエビデンスとなる情報が想定されている Web ページによって書き込まれたことを示すには，オリジン等との関係を示す必要がある．

よってメモリイメージのエビデンスとしての有効性を示すには，イメージ内の情報について以下が重要となる．

- オリジン，あるいはキー名を元に Web Storage に記憶されたバリューを機械的に検索可能であること．
- オリジン，キー及びバリューの各値が抽出可能であり，かつそれらの対応関係が明確に示せること

6.2 Web storage 内容獲得結果

Windows 上の各種 Web ブラウザの Web storage 使用後のメモリイメージから，記録内容の取り出しを試みた結果の表を表 1 に示す．○はエビデンスの取得に成功した項目，×は取得に失

ⓒ2012 Information Processing Society of Japan

敗した項目，△はエビデンス取得に成功したが，値がオリジナルから何らかの変換を施されている場合，あるいは抽出に何らかの知識が必要となることを示す．

Google Chrome に関して，Web storage 記録内容のメモリイメージからの獲得は，Session storage，Local storage の双方について失敗した．

また Internet Explorer に関して，Local storage 記録内容のメモリイメージからの獲得は一部に関して成功した．これはオリジン値の獲得については失敗したが，キー，およびバリュー値の獲得に成功したためで，探索時にはどちらかの値についての知識が無ければ機械的検索が不可能であることになる．Session storage については獲得に失敗しており，更なる検討，試行が必要である．

Mozilla Firefox に関して，Session storage 記録内容のメモリイメージからの獲得に成功している．これはオリジン，キー，バリューの各値が揃っており，かつ JSON 形式により明解にエンコードされていることから，エビデンスとして有効な形式での確認に成功した．一方 Local storage については，オリジンの値は一部変形されている．これについては変形規則が明解である．またキー，バリュー各値はデリミタなしで連結されており，キー，バリューどちらの値についても知識がなければ分離できないことから△としている．

表 1 storage 情報取得結果

browser	storage	origin	key	data
Firefox	local	△	△	△
	session	○	○	○
IE	local	×	○	○
	session	×	×	×
Chrome	local	×	×	×
	session	×	×	×

6.3 今後の課題

今回の実験を通して，Google Chrome の Web storage 記録のエビデンスの取得には失敗している．これに関し，Google Chrome 内部では，何らかの暗号処理を施しているものと推測される．当該ブラウザはオープンソースであることからソースコードを確認する必要がある．また Internet Explorer の Session storage 記録についてもエビデンス取得に失敗しており，当該ブラウザはソースコード以外の手段で確認する必要がある．

エビデンス取得に成功したブラウザについて，今回は仮想マシン上での実験であったが，実計算機上での実験による確認が必要である．また複数オリジンの情報が混在する状況で，メモリイメージ内にキー，バリュー値がどのように収容されるか，これらの値のメモリイメージ上での識別が可能かの確認が必要である．また，メモリイメージ上にエビデンスが残されるメカニズムについても，ソースコードの解析などを通じて理解する必要がある．

HTML 5 はスマートフォンやタブレットなどのモバイルデバイス上で動作するアプリケーションの記述言語として期待されており，本方式のモバイルフォレンジックへの応用を目指すためには，スマートフォンの実機上でのメモリイメージ取得方法についても検討を加え，実験を行う必要がある．

参考文献

1) 辻井 重男監修，特定非営利活動法人 デジタルフォレンジック研究会編集,「デジタル・フォレンジック辞典」株式会社日科技連出版社，2006年
2) Robin Berjon, et al., “HTML5 A vocabulary and associated APIs for HTML and XHTML W3C Candidate Recommendation 6 August 2013”, [Online] http://www.w3.org/TR/2013/CR-html5-20130806/, 6 August 2013.
3) D. Kristol, “HTTP State Management Mechanism”, RFC2109, Feb. 1997.
4) Ian Hickson, “Web Storage W3C Recommendation 30 July 2013”, [Online] http://www.w3.org/TR/2013/ REC-webstorage-20130730/, 30 July, 2013.
5) Murilo Tito Pereira, “Forensic analysis of the Firefox 3 Internet history and recovery of deleted SQLite records”, Digital Investigation: The International Journal of Digital Forensics & Incident Response archive, Volume 5 Issue 3-4, March, 2009, Pages 93-103.
6) Junghoon Oh, Seungbong Lee and Sangjin Lee, “Advanced evidence collection and analysis of web browser activity”, Digital Investigation: The International Journal of Digital Forensics & Incident Response archive, Volume 8, August, 2011, Pages S62-S70.
7) Donny Jacob Ohana et.al, “Do Private and Portable Web Browsers Leave Incriminating Evidence? A Forensic Analysis of Residual Artifacts from Private and Portable Web Browsing Sessions”, Security and Privacy Workshop (SPW), 2013 IEEE 23-24 May 2013, p135-142.
8) Ian Hickson, “HTML Standard”, [Online] http://www.whatwg.org/specs/web-apps/current-work/multipage/
9) Simon Pieters, “Differences from HTML4”, [Online] http://www.w3.org/TR/2013/WD-html5-diff-20130528/
10) European Network and Information Security Agency, “A Security Analysis of Next Generation Web Standards”, [Online] https://www.enisa.europa.eu/activities/Resilience-and-CIIP/critical-applications/web-security/a-security-analysis-of-next-generation-web-standards, 2011.
11) 一般社団法人 JPCERT コーディネーションセンター，“HTML5 を利用した Web アプリケーションのセキュリティ問題に関する調査報告書”, [Online]http://www.jpcert.or.jp/research/ HTML5-20131030.pdf
12) Ian Hickson, “HTML Microdata”, [Online] http://www.w3.org/TR/2013/NOTE-microdata-20131029/

ⓒ2012 Information Processing Society of Japan

13) Rik Cabanier, et al., “HTML Canvas 2D Context”, [Online] http://www.w3.org/TR/2013/CR-2dcontext- 20130806/
14) Ian Hickson, “HTML5 Web Messaging”, [Online] http://www.w3.org/TR/webmessaging/
15) Ian Hickson, “Web Workers”, [Online] http://www.w3.org/TR/2012/CR-workers-20120501/
16) Ian Hickson, “The WebSocket API”, [Online] http://www.w3.org/TR/2012/CR-websockets-20120920/
17) Ian Hickson, “Server-Sent Events”, [Online] http://www.w3.org/TR/2012/CR-eventsource-20121211/
18) A.Barch, “The Web Origin Concept”, RFC6454.
19) Top 5 Desktop, Table and Console Browsers from Jan 2013 to Jan 2014 | StatCounter Global Stats,(最終確認日: 2014 年 2 月 3 日), thttp://gs.statcounter.com/
20) AccessData FTK. [Online]. Available: http://www.accessdata.com/products/digital-forensics/ftk

付録 実験用コード抜粋

実験に用いた HTML 5/JavaScript コードの抜粋を以下に示す．コードは Local storage 実験用のものであるが，Session storage 実験についても同様である．

```
<script type="text/javascript">
      var KEY = "lsKey";
      var note = document.getElementById("note");
      storValue = "Gur Dhvpx Oebja Sbk Whzcf Bire Gur Ynml Qbt"
      localStorage.setItem(KEY, storValue);
      var text = rotThirteen(localStorage.getItem(KEY));

      if(text != null){
          note.value = text;
      }
```

付録 図 1 実験用コード抜粋

ⓒ2012 Information Processing Society of Japan